Homework 5: Pacman

Course: CS 221 Spring 2019
Name: Bryan Yaggi

Problem 1: Minimax

(a) Before you code up Pac-Man as a minimax agent, notice that instead of just one adversary, Pac-Man could have multiple ghosts as adversaries. So we will extend the minimax algorithm from class (which had only one min stage for a single adversary) to the more general case of multiple adversaries. In particular, *your minimax tree will have multiple min layers (one for each ghost) for every max layer.*

Specifically, consider the limited depth tree minimax search with evaluation functions taught in class. Suppose there are $n+1$ agents on the board, $a_0, \ldots, a_n$, where a_0 is Pac-Man and the rest are ghosts. Pac-Man acts as a max agent, and the ghosts act as min agents. A single depth consists of all $n+1$ agents making a move, so depth 2 search will involve Pac-Man and each ghost moving two times. In other words, a depth of 2 corresponds to a height of $2(n+1)$ in the minimax game tree.

Write the recurrence for $V_{minmax}(s,d)$ in math. You should express your answer in terms of the following functions: `IsEnd(s)`, which tells you if s is an end state; `Utility(s)`, the utility of a state; `Eval(s)`, an evaluation function for the state s; `Player(s)`, which returns the player whose turn it is; `Actions(s)`, which returns the possible actions; and `Succ(s,a)`, which returns the successor state resulting from taking an action at a certain state. You may use any relevant notation introduced in lecture.

$$V_{minmax}(s,d) = \begin{cases} \texttt{Utility(s)}, & \texttt{IsEnd(s)} \\ \texttt{Eval(s)}, & d = 0 \\ max_{a \in \texttt{Actions(s)}} V_{minmax}(\texttt{Succ(s,a)}, d), & \texttt{Player(s)} = a_0 \\ min_{a \in \texttt{Actions(s)}} V_{minmax}(\texttt{Succ(s,a)}, d), & \texttt{Player(s)} = a_1, \ldots, a_{n-1} \\ min_{a \in \texttt{Actions(s)}} V_{minmax}(\texttt{Succ(s,a)}, d-1), & \texttt{Player(s)} = a_n \end{cases}$$

(b) coding

Problem 2: Alpha-Beta Pruning

(a) coding

Problem 3: Expectimax

(a) Random ghosts are of course not optimal minimax agents, so modeling them with minimax search is not optimal. Instead, write down the recurrence for $V_{exptmax}(s,d)$, which is the maximum expected utility against ghosts that each follow the random policy which chooses a legal move uniformly at random. Your recurrence should resemble that of Problem 1a (meaning you should write it in terms of the same functions that were specified in 1a).

$$
V_{exptmax}(s,d) = \begin{cases} \texttt{Utility(s)}, & \texttt{IsEnd(s)} \\ \texttt{Eval(s)}, & d = 0 \\ max_{a \in \texttt{Actions(s)}} V_{exptmax}(\texttt{Succ(s,a)}, d), & \texttt{Player(s)} = a_0 \\ \sum_{a \in \texttt{Actions(s)}} \pi_{opp}(s,a) V_{exptmax}(\texttt{Succ(s,a)}, d), & \texttt{Player(s)} = a_1, \dots, a_{n-1} \\ \sum_{a \in \texttt{Actions(s)}} \pi_{opp}(s,a) V_{exptmax}(\texttt{Succ(s,a)}, d-1), & \texttt{Player(s)} = a_n \end{cases}
$$

$$
\pi_{opp} = \frac{1}{|\texttt{Actions(s)}|}
$$

(b) coding

Problem 4: Evaluation Function (Extra Credit)

So far, we've seen how MDP algorithms can take an MDP which describes the full dynamics of the game and return an optimal policy. But suppose you go into a casino, and no one tells you the rewards nor the transitions. We will see how reinforcement learning can allow you to play the game and learn its rules and strategy at the same time!

(a) coding

(b) Clearly describe your evaluation function. What is the high-level motivation? Also talk about what else you tried, what worked and what didn't. Please write your thoughts in `pacman.pdf` (not in code comments).

The overall strategy is as follows. If there are scared ghosts, chase the nearest scared ghost. If there are no scared ghosts but there are remaining capsules, chase the nearest capsule. If there are no scared ghosts and all capsules are gone, chase the nearest food. Features included the score and reciprocal of the distance to nearest objective. Distance is calculated using the `manhattanDistance` function in `util.py`. A penalty is added if a capsule is collected while a scared ghost is present.